REMARKS

INTRODUCTION

Claims 1-6 were previously pending and under consideration.

Claim 7 is added herein. Claim 7 is a method claim similar to claim 1 without "means".

Therefore, claims 1-7 are now pending and under consideration.

Claims 1-6 are rejected.

Claim 1 is amended herein.

No new matter is being presented, and approval and entry are respectfully requested.

CLARIFICATION OF REJECTION REQUESTED

If the Bannon reference is relied on for further rejection, Applicant respectfully requests the Examiner to specifically point out which portions of Bannon are proposed to correspond with which elements of the claims. The same four portions of Bannon are cited with respect to each of the numerous elements of the claims. Applicant cannot tell what in Bannon is supposed to correspond to an ODB, definition extraction, etc. It is respectfully noted that where applicable, the Examiner's findings should clearly articulate which portions of the reference support any rejection (MPEP § 2144.08). Furthermore, when a reference is complex or shows or describes inventions other than that claimed by the applicant, the particular part relied on by the Examiner must be designated as nearly as practicable (37 C.F.R. § 1.104(c)(2)). Finally, MPEP § 706.07 states that "a clear issue between applicant and examiner should be developed, if possible".

The following remarks are based on Applicant's best understanding of the rejection. However, if Bannon is again relied on to reject the claims, clarification of the proposed correspondences between Bannon and the individual elements of the present invention is respectfully requested.

REJECTIONS UNDER 35 USC § 102

In the Office Action, at pages 2-5, claims 1-6 were rejected under 35 U.S.C. § 102 as being anticipated by Bannon. This rejection is traversed and reconsideration is requested.

BACKGROUND

ODBs Different from RDBs

Some background and definitions are discussed to clarify features of the presently claimed invention that are not found in Bannon. Claims 1, 4, and 7 recite "changing over". The hyperdictionary.com indicates that to "change over" can mean to "change from one system to another or to a new plan or policy; 'We converted from 220 to 110 Volt' ", synonyms are to "convert, shift, switch". Furthermore, RDBs and ODBs have distinct differences. The Background of the present specification discusses these differences between ODBs and RDBs. Www.visionbase.co.uk/datasheets/vd21.pdf, also discusses these differences:

> Primarily, RDBMSs have been built around central server architectures, which are much the same as mainframe architectures. ODBMSs often assume a network of computers, with processing on the back or front end, as well as intermediate tiers, caching on each level, and clustering capabilities independent of type. In terms of computation model, although RDBMSs typically confine all processing to the SQL language and its operations (SELECT/PROJECT/JOIN and INSERT/UPDATE/DELETE), ODBMSs allow the use of host object languages like C++, Java, and Small talk *directly on the objects "in the database"; that is, instead of translating back and forth between application language structures (COBOL, C, etc.) and database structures (SQL). Application programmers can simply use the object language to create and access objects through the methods*. The [object] database [ODB] system maintains the persistence, integrity, recoverability, and con currency of those same objects.

Typical differences between an ODB and an RDB have also been summarized as:

1) representation of relationships
   ODB: relationship properties or reference attributes
   RDB: attributes with matching values, e.g., foreign keys

2) inheritance

ODB: built-in into model, e.g., derived(:) and EXTENDS

RDB: no built-in constructs

3) specification of operations

ODB: operations are part of the class specifications

RDB: implementation phase

See nlg3.csie.ntu.edu.tw/courses/ Database/slides/Dbase12.ppt. For further detailed discussion about differences between ODBs and RDBs, see also software.fujitsu.com/ en/Jasmine/ises2001_lecture.pdf. Although all of these properties may not be present in every particular instance of an ODB, these properties do highlight the general architectural difference between an ODB and an RDB or an object-oriented RDB hybrid (e.g. Bannon).

In sum, with an ODB, applications usually use ODQL or the like to access the stored objects, an RDB is not used, and the ODB handles both management and storage of persistent objects.

As discussed in detail below, the presently claimed invention differs fundamentally from Bannon in that the presently claimed invention is for changing over an RDB system to an analogous, corresponding, or correlated ODB system, whereas Bannon involves only one RDB-based object storage system that is not changed over or converted to another system such as an ODB system.

Present Invention; ODB

In general, the presently claimed invention relates to changing over or converting a relational database (RDB) to an object database (ODB). A change over or conversion involves creating a new system (e.g. ODB) analogous to a previous system (e.g. RDB) but with a different base technology for the new system. The change over is accomplished in part by extracting definition information from the RDB and using it to create the ODB. The new changed over system (ODB) is by nature autonomous and can stand alone without the old system while reflecting the relations of the data stored in the RDB.

With reference to the claims, claims 1, 4, and 7 recite "changing over an existing relational database to an object database", "extracting RDB definition information from an RDB

repository", "where the relational database is a transition object to be transitioned to the object database", and "creating an ODB repository describing therein definition information of the object database associated with the RDB definition information in accordance with the [extracted] RDB definition information ... and for creating correlation information repository defining mutual relationship between the RDB definition information and ODB definition information". In other words, the definition of the RDB is extracted and used to create a mutually related or correlated ODB.

Bannon; RDB or RDB Hybrid

The purpose of Bannon is to create a portable platform-independent persistent object storage system for at least one application written in object-oriented language(s). "Portable" means that the same system can be compiled and executed on different types of platforms. Bannon uses a Data Definition Language (DDL) translator, an Object Management System (OMS), an Object Translation System (OTS), and a Persistent Object Storage Server (POS Server). A user must register classes to be used using the DDL translator, where the DDL is an extended class definition language. The DDL translator is a C preprocessor that inputs object descriptions to obtain storage details such as size, padding, alignment, etc., which is used by the OTS to translate between primary and secondary storage. The OMS provides an interface for database-like operations and for automatic or implicit retrieval of objects from the RDB. Applications access persistent objects using a special data type called the ZG_PTR. The POS provides stable long-term storage via its RDB using embedded SQL.

Fundamental Differences

In sum, Bannon is not an ODB system but is rather a three-tier object storage system with an application tier, an object management tier, and an RDB storage tier. Bannon does not convert or change over one type of system to another, for example converting data or applications. These differences are discussed in detail below with reference to the claims.

BANNON DOES NOT DISCLOSE AN ODB

With an ODB as recited in the present claims, object management is performed within an ODB. Applications can control storage of objects directly, for example using Object SQL (OSQL) without the need for a middle management layer as in Bannon. Bannon discloses an RDB-object system. With Bannon, there is no ODB to perform both object management and storage, rather the OMS manages objects (e.g. names, references, primary-secondary memory, etc.) and the RDB manages persistent storage. Thus Bannon lacks an ODB as recited in the present claims. As discussed above, an "ODB" is recognized in the art of object persistence as being a distinctly different type of system than a multi-layer RDB-based system as in Bannon. For example, in Bannon, applications store objects to the same POS (col. 8, lines 20+), where relational tables defined in the RDBMS store persistent objects created by applications. This is distinctly different from the type of object storage provided by an ODB. Furthermore, an ODB is an integrated persistent storage system accessed directly by objects. In the system of Bannon, the RDB and the object storage management are separate; applications do not directly access objects (for example using OSQL), but rather use the middle tier to indirectly access persistent objects. In sum, the differences between an ODB and the system of Bannon are significant, and although Bannon provides persistent object storage, it does so in a completely different manner than an ODB as created by the changing over of claims 1, 4, and 7.

BANNON DOES NOT CONVERT OVER TO AN ODB

Claim 1, for example, recites a change over where a new ODB is created based on an RDB. With the present invention, the creation of a new ODB system obviates the need for an RDB; the new ODB now provides the persistent storage and the RDB becomes superfluous and disposable. In contrast, with Bannon, there is only one system and persistent storage management occurs at least in part within an RDB. Bannon requires that the RDB be maintained. In Bannon, neither the OMS nor the OTS provide conversion from an RDB system to a corresponding ODB.

The change over in claims 1, 4, and 7 is accomplished in part by extracting from an RDB repository definition information of the relational database. Bannon does not disclose this feature. In Bannon, the definition of the RDB is driven by the DDL class definitions provided by the programmer. The RDB simply reflects the necessities of the classes of objects being stored. Because there is no conversion or change over from an RDB to a corresponding ODB system,

there is no need to extract the definition of the RDB. Bannon provides conversion of a same object instance between primary and secondary storage formats, and between the RDB format and the object layout in an application. This is not the same as converting from one type of system to another based on extracted definition information. The definition information of Bannon is provided by way of a programmer's DDL files used during initial setup.

CLAIMS 2 AND 5: DATA CONVERSION IN ACCORDANCE WITH CORRELATION INFORMATION NOT DISCLOSED BY BANNON

Claim 2, for example, recites "converting data of the relational database into the object database in accordance with the correlation information". The Merriam Webster Dictionary indicates that "convert" can be used to indicate "to change from one form or function to another". As discussed above, Bannon does not discuss converting from an RDB to an ODB. Therefore, Bannon cannot convert data from an RDB into data of a correlated ODB.

CLAIMS 3 AND 6: APPLICATION CONVERSION NOT DISCLOSED BY BANNON

Claim 3, for example, recites "converting an application program described in a relational database based language into an application program described in an object database based language in accordance with the correlation information [correlating the RDB and ODB]". There is no discussion in Bannon of changing the form or function of an application. Applications are simply written using the ZG_PTR construct. An application can be ported to another platform per the usual procedure of manually recompiling the application and so on.

CONCLUSION

There being no further outstanding objections or rejections, it is submitted that the application is in condition for allowance. An early action to that effect is courteously solicited.

Finally, if there are any formal matters remaining after this response, the Examiner is requested to telephone the undersigned to attend to these matters.

If there are any additional fees associated with filing of this Amendment, please charge the same to our Deposit Account No. 19-3935.

Respectfully submitted,

STAAS & HALSEY LLP

Date: 19 Dec 2003

By: James Strom

James T. Strom
Registration No. 48,702

1201 New York Avenue, NW, Suite 700
Washington, D.C. 20005
Telephone: (202) 434-1500
Facsimile: (202) 434-1501